(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平8－242286

(43)公開日　平成８年(1996)９月17日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｈ０４Ｍ　3/00			Ｈ０４Ｍ　3/00	Ｄ	
3/22			3/22	Ｚ	

審査請求　未請求　請求項の数７　ＯＬ　（全 18 頁）

(21)出願番号　特願平7－44569

(22)出願日　平成７年(1995)３月３日

特許法第30条第１項適用申請有り　1994年９月26日～９月29日　社団法人電子情報通信学会主催の「電子情報通信学会1994年秋季大会－ソサイエティ先行大会－」において文書をもって発表

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　浜田　健生
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(72)発明者　伊勢田　衡平
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(74)代理人　弁理士　伊東　忠彦

最終頁に続く

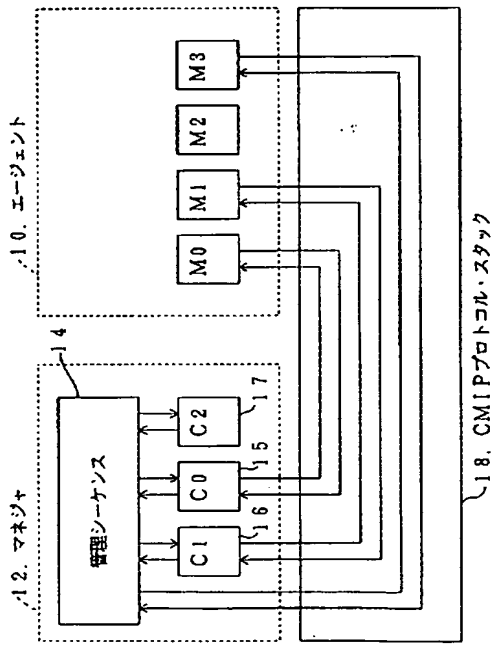
(54)【発明の名称】　通信網管理制御方式

(57)【要約】

【目的】　本発明は通信管理制御方式に関し、マネジャが実行する管理シーケンスを高速化し、キャッシュ管理を最適化してキャッシュ・コヒーレンシを確保すると共に管理トラヒックの増大を防止し、キャッシュをエージェントの障害復旧に有効利用することを目的とする。

【構成】　通信ネットワークの管理主体１２から代理主体１０にコマンドを送り、代理主体で通信ネットワークの管理情報のうちの管理対象を操作する通信網管理制御方式において、管理主体上に管理情報を格納するキャッシュ１５～１７を設け、キャッシュを管理情報のキャッシュ・コヒーレンシ要求度に対応した属性クラスに基づいて分割する。

本発明の構成図



1

【特許請求の範囲】

【請求項１】　通信ネットワークの管理主体から代理主体にコマンドを送り、上記代理主体で通信ネットワークの管理情報のうちの管理対象を操作する通信網管理制御方式において、

上記管理主体上に管理情報を格納するキャッシュを設け、

上記管理情報の属性をキャッシュ・コヒーレンシ要求度に対応してクラス分けしたことを特徴とする通信網管理制御方式。

【請求項２】　前記管理情報の定義から前記属性クラスを得て、属性クラス毎のキャシュ領域の確保及び初期値の設定を実行する初期化手段を自動生成することを特徴とする請求項１記載の通信網管理制御方式。

【請求項３】　前記管理情報の定義から前記属性クラスを得て、属性クラス毎のコヒーレンシ・プロトコルを実行するインタフェース処理手段を自動生成することを特徴とする請求項１又は２記載の通信網管理制御方式。

【請求項４】　前記属性クラスに基づいて分割されたキャッシュを上記属性クラスに基づいて管理し、かつ属性クラスに対応したコヒーレンシ・プロトコルを適用してキャッシュ・コヒーレンシを確保することを特徴とする請求項１乃至３のいずれかに記載の通信網管理制御方式。

【請求項５】　前記属性クラスに基づいて分割されたキャッシュ夫々をリースト・リーセントリー・ユースト・アルゴリズムにより管理することを特徴とする請求項１乃至４のいずれかに記載の通信網管理制御方式。

【請求項６】　前記代理主体に、自らと接続されている管理主体の情報を記録する不揮発性記録手段を有することを特徴とする請求項１記載の通信網管理制御方式。

【請求項７】　前記代理主体の障害復帰時に、前記不揮発性記録手段に記録された管理主体上のキャッシュから上記代理主体の管理情報を復旧することを特徴とする請求項６記載の通信網管理制御方式。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は通信網管理制御方式に関し、オブジェクト指向のネットワークの通信網管理制御方式に関する。最近、オブジェクト指向の考えで統一されるＴＭＮ（Telecommunication Management Network）と呼ばれる通信網管理制御方式が開発されている。

【０００２】通信の広帯域化、高信頼化への要求は通信網のネットワーク管理の発展を促し、ＩＴＵ－Ｔ（ＣＣＩＴＴ）においてもネットワーク管理の国際標準としてＴＭＮに関する勧告Ｍ．３１００が行われている。高信頼かつ高効率なネットワーク管理を行う為には、網管理に要する処理上の負荷、通信上の遅延に対処するような通信網管理制御方式が要求されている。

【０００３】

2

【従来の技術】ＴＭＮに基づく通信網管理では、通信機器の状態や通信網の接続状態等の管理情報はＭＯ（Managed Object）と呼ばれるオブジェクトとして表現される。通信管理制御は、マネジャ（Manager）と呼ばれる管理主体がＣＭＩＰ（Common Managed Information Protocol）と呼ばれる通信プロトコルを用いて、エージェント（Agent）と呼ばれる代理主体にコマンドを送り、エージェントはそのコマンドの内容に従って管理対象のＭＯを操作することによって行われる。

【０００４】エージェント上でのＭＯの集合は管理情報ベース（ＭＩＢ）と呼ばれる。また、ＭＯの属性（Atribute）は管理情報を構成する要素であり、例えばスイッチのオン／オフ状態等の物理的な意味を持っている。例えは、マネジャはエージェントに対してＣＭＩＰのコマンドを送ることにより、ＭＯの属性値を読み出し（Ｍ－ＧＥＴ）、又は書き込む（Ｍ－ＳＥＴ）。通信網管理は通常このようなＣＭＩＰコマンドのシーケンス（管理シーケンス）によって行われる。

【０００５】

【発明が解決しようとする課題】従来、ＴＭＮによる通信管理を大規模なネットワークに適用した場合、マネジャとエージェントとの間の網内遅延が大きくなり、この網内遅延にＣＭＩＰのプロトコル処理に要する遅延を加えると、遅延時間が数十msecに達する場合があった。このため、トラヒック状況や異常監視等のために、マネジャがエージェント上のＭＯ属性情報に頻繁にアクセスする必要のある管理シーケンスは実行に大きな時間がかかり、高速化が困難であるという問題があった。

【０００６】一方、キャッシュを用いて網内遅延を小さくする手法が以前から知られている。キャッシュを用いることにより、アクセス時間、特に属性値の読み出しに要する時間を大幅に短縮できる。しかし、キャッシュを用いる場合、各キャッシュ間の内容の一致（コヒーレンシ）を保証する必要があり、コヒーレンシ・プロトコルを用いて常にコヒーレンシを保つ必要があった。そのためにキャッシュの数の増加に伴ってコヒーレンシ・プロトコルによるメッセージが増加し、このことにより管理トラヒックが増大しネットワークの負荷が増え、通信網管理の運用方法によってはネットワークが輻輳状態に陥る危険性があった。つまり、従来の技術ではキャッシュを持つ事の利点と、ネットワークの管理トラヒックの増大を抑えるという目標を両立させることが困難であり、このため通信網管理においてキャッシュ技術は広く用いられるに到らなかった。

【０００７】また、エージェントに障害が起こった場合に対処するため、ＭＯの属性値をデータベースに格納し、障害復旧時にデータベース上の値を用いてＭＯの属性値を復帰する方法がとられる。また、同じ目的でＭＯの属性値の変更時に複数のマシンにコピーすることによってＭＯ属性値の保存を可能にする方法が知られてい

る。より具体的には、複数のホスト上にデータベースを配置して、一箇所のデータベースでＭＯ属性値に変更があれば、他のデータベースにも同じ値をコピーする。この方法によれば、障害時のフォールト・トラレンシーは向上するが、コピーはマネジャ上のキャッシュとは独立であり、マネジャ上の管理アプリケーションによりキャッシュとして用いられることはなく、このため、通信網管理システムの高速化には直接貢献せず、またキャッシュを独立に設けた場合は同じ情報をデータベースに重複して持つことになり、効率が悪いという問題があった。

【０００８】本発明は上記の点に鑑みなされたもので、マネジャが実行する管理シーケンスを高速化し、キャッシュ管理を最適化してキャッシュ・コヒーレンシを確保すると共に管理トラヒックの増大を防止し、キャッシュをエージェントの障害復旧に有効利用する通信管理制御方式を提供することを目的とする。

【０００９】

【課題を解決するための手段】請求項１に記載の発明は、通信ネットワークの管理主体から代理主体にコマンドを送り、上記代理主体で通信ネットワークの管理情報のうちの管理対象を操作する通信網管理制御方式において、上記管理主体上に管理情報を格納するキャッシュを設け、上記管理情報の属性をキャッシュ・コヒーレンシ要求度に対応してクラス分けする。

【００１０】請求項２に記載の発明は、前記管理情報の定義から前記属性クラスを得て、属性クラス毎のキャシュ領域の確保及び初期値の設定を実行する初期化手段を自動生成する。請求項３に記載の発明は、前記管理情報の定義から前記属性クラスを得て、属性クラス毎のコヒーレンシ・プロトコルを実行するインタフェース処理手段を自動生成する。

【００１１】請求項４に記載の発明は、前記属性クラスに基づいて分割されたキャッシュを上記属性クラスに基づいて管理し、かつ属性クラスに対応したコヒーレンシ・プロトコルを適用してキャッシュ・コヒーレンシを確保する。請求項５に記載の発明は、前記属性クラスに基づいて分割されたキャッシュ夫々をリースト・リーセントリー・ユースト・アルゴリズムにより管理する。

【００１２】請求項６に記載の発明は、前記代理主体に、自らと接続されている管理主体の情報を記録する不揮発性記録手段を有する。請求項７に記載の発明は、前記代理主体の障害復帰時に、前記不揮発性記録手段に記録された管理主体上のキャッシュから上記代理主体の管理情報を復旧する。

【００１３】

【作用】請求項１に記載の発明においては、管理主体にキャッシュを設けることにより、管理主体が実行する管理シーケンスを高速化することができ、管理情報の属性をキャッシュ・コヒーレンシ要求度に基づいてクラス分けすることによりキャッシュ管理の最適化をはかることが可能となる。

【００１４】請求項２に記載の発明においては、管理情報の定義から初期化手段が自動生成され、この初期化手段によって管理主体に属性分割キャッシュが確保され、かつ、その初期化が行われ、属性分割キャッシュを使用しない装置との相互運用性を確保できる。

【００１５】請求項３に記載の発明においては、管理情報の定義からインタフェース処理手段が自動生成され、このインタフェース処理手段によって属性クラス毎のコヒーレンシ・プロトコルが実行されるため、属性キャッシュ夫々に最適のコヒーレンシ・プロトコルが実行可能となる。

【００１６】請求項４に記載の発明においては、属性分割キャッシュが夫々に最適のコヒーレンシ・プロトコルを用いてキャッシュ・コヒーレンシが確保されるため、ネットワークの管理トラヒックの増大を防止してネットワークの負荷の増大を抑制できる。

【００１７】請求項５に記載の発明においては、属性分割キャッシュをリースト・リーセントリー・ユースト・アルゴリズムを用いて管理するため、キャッシュ領域が増大化することを避けることができ、メモリの有効利用が可能となる。請求項６に記載の発明においては、不揮発性記録手段を有するため、代理主体と接続されている管理主体の情報を代理主体の障害時に確保できる。

【００１８】請求項７に記載の発明においては、代理主体の障害復旧時に、これと接続されていた管理主体のキャッシュから管理情報を復旧できるため、キャッシュを代理主体の障害復旧に有効利用できる。

【００１９】

【実施例】図１は本発明の構成図を示す。同図中、エージェント１０は管理情報属性を物理的性質やキャッシュ・コヒーレンシの要求度の高さにより分割し、属性クラスＡ～Ｄの管理情報Ｍ０～Ｍ３に分けて管理している。

【００２０】マネジャ１２は管理シーケンス１４を有すると共に、キャッシュ１５，１６，１７を有している。キャッシュ１５～１７はエージェント１０における属性クラスＡ～Ｃの管理情報Ｍ０，Ｍ１，Ｍ２夫々に対応して設けられており、属性クラスＡ～Ｃの管理情報Ｃ０，Ｃ１，Ｃ２を格納する。ＣＭＩＰプロトコル・スタックは上記エージェント１０とマネジャ１２との間を接続しており、両者間でＣＭＩＰコマンドが送受されてエージェント１０の管理情報がマネジャ１４に送られる。

【００２１】上記の属性クラスＡ～Ｄについて説明する。属性クラスＡは、一時的なキャッシュの不一致が許される属性であり、ＣＭＩＰコマンドの“ＧＥＴ”でマネジャから読み出し操作のみが許されるＧＥＴ－ＯＮＬＹ属性である。この属性としては監視情報のＭＯ、パスに関するコスト情報のＭＯ、網の統計情報のＭＯ等がある。

【００２２】属性クラスＢは、厳密なコヒーレンシが必

要な属性であり、ＣＭＩＰコマンドの“ＧＥＴ－ＲＥＰＬＡＣＥ”でマネジャから読み出し又は書き換え操作により属性値の設定が可能なＧＥＴ－ＲＥＰＬＡＣＥ属性である。この属性としては、ドメイン境界のクロスコネクト情報のＭＯ等がある。

【００２３】属性クラスＣは、ＧＥＴ－ＯＮＬＹ属性であり、ＣＭＩＰコマンドのＭ－ＣＲＥＡＴＥを用いたＭＯの生成時に固定され、ＭＯの存在期間において変化する可能性のない固定的な属性である。属性クラスＤは、一旦生成されると読み出されることは殆どなく、キャッシュを設ける必要性の無い属性である。

【００２４】ある属性がどの属性クラスに属するかの分割は、図２に示す如きＧＤＭＯ定義に与えられる類別（ＧＥＴ，ＧＥＴ－ＲＥＰＬＡＣＥ）から大まかに可能であり、その属性の物理的な意味からどのクラスに属するかを判別できる。図２においては、ステップＳ１００，Ｓ１１０からoperationalStateＧＥＴ－ＯＮＬＹ属性であり、物理的な意味から属性クラスＡである。ステップＳ８０，Ｓ９０からadministrativeState がＧＥＴ－ＲＥＰＬＡＣＥ属性であるので属性クラスＢである。ステップＳ６０，Ｓ７０，Ｓ１２０～Ｓ１９０からcrossConnection ＩＤ，signal Type ，from Termination、to Termination、directionality夫々がＧＥＴ－ＯＮＬＹ属性であり、物理的な意味から属性クラスＣである。図２の例では属性クラスＤは存在しない。

【００２５】上記の属性クラスＣの属性は一旦属性値が与えられると変化することはないので、図１のマネジャ１２のキャッシュ１７の管理情報Ｃ２とエージェント１０の管理情報Ｍ２との一致は常に保たれる。属性クラスＡ，Ｂの属性（管理情報Ｍ０，Ｍ１）は夫々マネジャ１２のキャッシュ１５，１６に格納され、ＣＭＩＰプロトコル・スタック１８のコヒーレンシ・プロトコルによりコヒーレンシが保たれる。属性クラスＤの属性（管理情報Ｍ３）はＣＭＩＰプロトコル・スタック１８のＭ－ＧＥＴ，Ｍ－ＳＥＴを用いて属性値の読み出し、変更が可能である。

【００２６】図１に戻って説明するに、属性クラスＣのＭＯについては、マネジャ１２とエージェント１０との間の通信が必要なのは初期値の設定の場合のみであるから、通常のアクセスにおいてはキャッシュ・コヒーレンシの維持のためのＣＭＩＰを必要としない。属性値の設定後読み出しが殆ど起こる可能性の無い属性クラスＤについては、キャッシュを適用する必要性は少ないので、キャッシュを用いない。

【００２７】マネジャ１２において管理シーケンス１４の実行時に管理情報Ｍ０，Ｍ１，Ｍ２に対してはそれぞれのキャッシュ１５～１７の管理情報Ｃ０，Ｃ１，Ｃ２をアクセスし、管理情報Ｍ３に対しては直接ＣＭＩＰのコマンドを用いて属性値の制御を行うことにより、管理シーケンス１４の実行を高速化することができる。管理情報Ｃ０，Ｃ１に対してはそれぞれのコヒーレンシ要求に応じたプロトコルを適用する。すなわち、属性クラスＡ、属性クラスＢについてはキャッシュと独立にエージェント１０上ＭＯの属性値が変化する可能性があり、コヒーレンシ・プロトコルにより両者の一致をとる必要がある。ＭＯの属性値が変化する理由として、エージェント１０の自立的な動作による変更、あるいはまた複数のマネジャが通信網内に存在する場合に、他のマネジャによるＭ－ＳＥＴの結果変化することなどが考えられる。

【００２８】次にコンパイル時のインタフェース処理部プログラムの自動生成について説明する。図３はＴＭＮ通信網管理制御に適用した場合のシステムフローチャートを示す。ここで、管理情報Ｍ０～Ｍ３，Ｃ０～Ｃ２は図１と対応している。マネジャ１２上で実行される管理シーケンス１４はＣ＋＋のような高級言語とＩＤＬのようなインタフェース記述言語を用いて記述される。一方、ＣＭＩＰによるＭＯへのアクセスのインタフェースはＧＤＭＯ（Guideline for the Definition ofManaged Object）により規定されており、ＩＤＬコンパイラ２２は通信網管理シーケンスからアクセスされるＭＯのＧＤＭＯ記述２０を参照しながら、マネジャ１２上のインタフェース処理部プログラム（インタフェース処理手段）であるstub.c、Manager 起動時の初期化プログラム（初期化手段）である manager__init.c、エージェント上のインタフェース処理部プログラム（インタフェース処理手段）であるskeleton.c Agent起動時の初期化プログラム（初期化手段）である agent__init.cを自動的に生成する。

【００２９】ＧＤＭＯ記述による図２のようなＭＯ記述において通常与えられるＧＥＴ，ＧＥＴ－ＲＥＰＬＡＣＥのような属性分類のみではコヒーレンシ要求度に関して十分な情報を与えないので、図５に示す如くコメント文の中に、pragma文でＩＤＬコンパイラ２２への指示を埋め込むことにより実現する。ＩＤＬコンパイラ２２は”－”で始まるコメント文の中にpragma文を見つけると、その指示に従ってマネジャ側においてキャッシュ領域の確保、コヒーレンシ・プロトコルの選択を行い、以後のキャッシュへのアクセス方法を決定する。具体的にはキャッシュ領域の確保、初期値の設定を行うプログラムを manager__init.cに出力し、キャッシュへのアクセスを行うプログラムをstub.cに出力する。同じ情報を用いてエージェント１０側で使用するプログラムについても、それぞれの属性クラスについて対応するプロトコルを用いるので、同様に初期化プログラム agent__init.c、インタフェース処理部skeleton.cを出力する。

【００３０】このようなpragma文による属性クラスの指定は、通信管理シーケンスの移植性、及びキャッシュを使用しない通常のＩＤＬコンパイラ２２による通信網管理シーケンスのコンパイルに何らの害も及ぼさない。理由としてまず通信網管理シーケンスを実行するプログラ

ムから見た場合、キャッシュの存在は透明であり意識する必要がなく、プログラムの実行結果はキャッシュの有無に関わらない。またＩＤＬコンパイラ２２から見た場合、pragma文を理解しないコンパイラにとっては単なるコメント文と同じであり、キャッシュを使用することなく通常のＣＭＩＰの使用によって通信網管理シーケンスを実現するからである。

【００３１】図４はＩＤＬコンパイラ２２の処理構成図を示す。同図中、ＩＤＬ文法解析部２５は通信網管理シーケンス１４内のインタフェース部ＩＤＬ記述１４ａの解析を行って属性クラス別コード要素生成部２６に供給する。また、ＧＤＭＯ文法解析部２７はＧＤＭＯ記述２０を解析し、この解析結果を用いて属性要素の属性クラス弁別部２８で属性クラスを弁別し属性クラス別コード要素生成部２６に供給する。

【００３２】属性クラス別コード要素生成部２６ではＩＤＬ解析結果及び属性クラス弁別結果に基づいて属性クラス別コード要素を生成する。マネジャ側コード要素生成部３０は上記属性クラス別コード要素からマネジャで使用するプログラムstub.c及び manager__init.c３１を生成し、エージェント側コード要素生成部２は上記属性クラス別コード要素からエージェントで使用するプログラムskeleton.c及びagent __init.cを生成する。

【００３３】ここで、図５に示すpragma文を補ったＭＯ定義と、これに対応して図６に示すＩＤＬ定義が与えられた場合について説明する。図６のステップＳ５２０に用いられている__objectInstance、__ObjectIdなどのタイプのデータ構造の定義は、ＩＴＵ－Ｔのドキュメントで定義されている。ＩＤＬコンパイラ２２は、図６のＩＤと定義の文法解析を行い、各属性に対応して以下のようなコード要素をstub.c及びskeleton.cの一部として生成する。例えば、図５のステップＳ２６０の属性crossConnectionId に対してはＧＥＴ－ＯＮＬＹ属性であるので、

```
ObjectId crossConnection__get __crossConnectionId();
```

という関数を生成し、一方ステップＳ２７０の属性administrativeState に対しては

```
enum crossConnection__get __administrativeState();
void crossConnection__set __administrativeState();
```

という関数を生成する。通常のＣＭＩＰを用いるシーケンスにおいては、マネジャ１２からの管理シーケンス中において属性administrativeState の値を参照する場合、

```
adminState= crossConnection__get __administrativeState(o,ev);
```

により得ることができ、一方属性の値を設定する場合

```
crossConnection__set __administrativeState(o,ev,adminStateValue)
```

のように行う。マネジャ１２側で起動されるそれぞれの関数はstub.cに含まれ、ＩＤＬコンパイラ２２によって生成されるコード要素である。ここでｏはエージェント上のＭＯを参照するポインタであり、evはマネジャプロセスの環境変数を表わすポインタである。エージェント側ではマネジャ側で起動された crossConnection__get __administrativeState()，crossConnection__set __administrativeState() に対応して同名の関数を起動し、ＭＯの実際の属性値の参照及び更新を行う。ここで、エージェント上で起動されるそれぞれの関数はskeleton.cに含まれ、ＩＤＬコンパイラ２２によって生成されるコード要素である。

【００３４】次に、キャッシュ管理について説明する。まず、キャッシュ領域の確保については、本発明の方式ではキャッシュ領域は通信網管理シーケンス１４を実行するプログラムのデータ領域にとられるので、通常のパーソナル・コンピュータ、ワークステーションのオペレーティング・システム（ＯＳ）において容易に実現され、何ら特殊なハードウェアないしＯＳを必要としない。また、キャッシュ領域の実現にあたり、キャッシュされる各ＭＯの属性とメモリ上の領域の対応があればどのような形態でも良く、通常のメモリを用いて容易に実現される。

【００３５】すなわち、具体的には通信網管理シーケンス１４からのＭ－ＣＲＥＡＴＥ要求によってエージェント１０上にＭＯインスタンスが生成されるのと並行してマネジャ１２上キャッシュ内にＭＯ属性対応のキャッシュが確保される。又はＭ－ＧＥＴ／Ｍ－ＳＥＴ要求によってマネジャ側からエージェントへのＭＯへのアクセス要求が生じた時点でキャッシュが確保され、同じく通信網管理シーケンスからのＭ－ＤＥＬＥＴＥ要求によってエージェント１０上ＭＯインスタンスが消去されるのと並行してキャッシュ内のＭＯ属性対応部分が消去され、あるいはキャッシュ内データが使用されなくなるとキャッシュ管理シーケンスによってＬＲＵ (Least Resently Used)などのアルゴリズムを用いて属性対応のキャッシュ内領域は解放される。

【００３６】図７は、図５において示したcrossConnectＭＯ属性のメモリ上のキャッシュ実現例を示している。図７において、crossConnect__Cache(o)は当該ＭＯのキャッシュ領域を指すポインタである。ここで、ｏは、ＭＯを参照するためのポインタであり、関数crossConnect__Cache() はハッシュ法などの方法によって当該ＭＯに対応するキャッシュ中の領域を計算する関数である。また図中 crossConnectionId, administrativeState などの領域はそれぞれの属性値を格納し、AdministrativeClass.Ｃ, AdministrativeClass.Ｂなどの領域は、それぞれの属性クラスを示している。またそれぞれの属性値は図中に示される属性番号〔１〕－〔７〕によって一意に指定することができる。属性クラス値は使用すべきコヒーレンシ・プロトコルを指定しており、通信網管理シー

ケンス１４から属性値の更新要求が起こると、マネジャ機能を実現している通信網管理システムは属性クラス値に従って適切なコヒーレンシ・プロトコルを起動することによりエージェント１０上のＭＯ属性値とキャッシュ上の内容の一致を保つ。すなわち、キャッシュの存在は通信網管理シーケンス１４の使用者からは透明になっている。

【００３７】また、図８はメモリ上のキャッシュの異なる実現例である。属性番号を付された各属性はポインタを介して間接的にアクセスされることにより、図７とは異なり、各属性を表わすデータはメモリ上の任意の場所に置くことが可能であり、またマネジャ１２側で使用しない属性（〔３〕，〔４〕）についてはキャッシュしない、という選択が可能となる。

【００３８】ここで、各属性クラスで使用するコヒーレンシ・プロトコルについて説明する。属性クラスＡでは、任意の二つのキャッシュの内容が一致するまでの時間がある一定値（τ）以下となるように制御し、具体的には定期的（τ以下）にエージェント側からマネジャ１２側のキャッシュ１５へブロードキャスト（Broadcast）を行い、キャッシュ１５に置かれている属性値を更新する。このコヒーレンシ・プロトコルをプロトコル（Ａ）と呼ぶ。

【００３９】属性クラスＢに対しては厳密なコヒーレンシを確保する必要があるので、キャッシュ１６を更新するためにクロックを用いる標準的なコヒーレンシ・プロトコルを適用する。この時、エージェント側からマネジャ側に、ＭＯの属性値の変化を通知する必要があるので、ＣＭＩＰにおけるＭ－Event-Reportのような機能を利用することができる。このコヒーレンシ・プロトコルをプロトコル（Ｂ）と呼ぶ。

【００４０】属性クラスＣに対しては、キャッシュの内容の一致は常に保たれているので、Manager-Agent 間の通信は必要なく、キャッシュの値を参照するのみで良い。次にキャッシュ１５～１７の初期設定について説明する。マネジャ１２とエージェント１０との間のＣＭＩＰのアクセスを最小限にするために、コンパイル時に知られている情報を用いて可能な限り、manager __init.c においてキャッシュ１５～１７を初期化する。

【００４１】まず、属性クラスで分割したキャッシュである属性分割キャッシュを使用し、かつ属性分割キャッシュを使用しない通常のＣＭＩＰに従って動作する網管理装置と相互運用性（Interoperability）を確保するためには、マネジャ側、エージェント側において適切な初期化操作が必要であり、ＩＤＬコンパイラ２２はそのような初期化ルーチンをそれぞれmanager __init.c, agent __init.cとして出力する。

【００４２】図９はマネジャ１２の初期化ルーチンのフローチャートを示す。同図中、まずステップでエージェント側に属性分割キャッシュ及び属性分割更新プロトコルを備えるか否かを問い合わせ、ステップＳ７１０でエージェント側からの上記問い合わせ結果を判別する。問い合わせ結果がＹＥＳの場合はステップＳ７２０に進んで属性分割キャッシュ及びプロトコル（Ａ），（Ｂ）等の属性分割更新プロトコルの適用を行い、問い合わせ結果がＮＯの場合はステップＳ７３０に進んで通常のＣＭＩＰを用いる属性更新を行う。

【００４３】図１０はエージェント１０の初期化ルーチンのフローチャートを示す。同図中、ステップＳ７５０ではマネジャ側からの属性分割キャッシュ及び属性分割更新プロトコルを備えるか否かの問い合わせを受け取り、ステップＳ７６０でキャッシュの使用状態を表現するCache ＭＯの属性を調べることにより属性分割キャッシュが使用可か否かをチェックする。

【００４４】次にステップＳ７７０で上記のチェック結果を問い合わせを行ったマネジャに送り返すと共に、ステップＳ７８０で上記チェック結果を判別し、属性分割キャッシュが使用可の場合にはステップＳ７９０でエージェント側からのブロードキャスト送信を可能とするためのイベントスケジューラ（event __scheduler)の初期化を行う。これは属性値変更時においては、プロトコル（Ａ），（Ｂ）のいずれにおいてもエージェント側からＭＯの属性値変化をマネジャ側にブロードキャスト送信する必要があり、それを可能とするためにエージェント１０のプロセスのevent __scheduler を初期化し、ＣＭＩＰのEventNotification の機能を用いて、属性値の変化をマネジャ側に通知するようにしている。

【００４５】次にキャッシュのアクセスについて説明する。図１１はマネジャ１２が実行するキャッシュアクセスのフローチャートを示す。プロトコル（Ａ）の場合はＧＥＴ操作及びＳＥＴ操作において、プロトコル（Ｂ）の場合はＧＥＴ操作において、マネジャからキャッシュをアクセスすることになる。

【００４６】同図中、まずステップＳ８００でハッシュ関数などを用いて、対応するＭＯがキャッシュ内に存在するか否かを確認する。対応するＭＯがキャッシュ内に有ればステップＳ８１０でキャッシュ内の対応するＭＯにアクセスし、無ければステップＳ８２０でキャッシュ管理シーケンスを起動し、ＣＭＩＰプロトコルによって対応するＭＯをキャッシュに書き込んでキャッシュ内容を更新する。この後、ステップＳ８３０においてキャッシュ夫々の属性クラスに応じたコヒーレンシ・プロトコルつまりプロトコル（Ａ），（Ｂ）を起動することによりキャッシュコヒーレンシを確保する。

【００４７】次にキャッシュ領域の管理について説明する。キャッシュはマネジャ１２のメモリ上に確保されるが、メモリ領域に制限がある場合、アクセスする可能性があるすべてのＭＯ属性をキャッシュにおくのは困難であるので、ＬＲＵ，ＣＬＯＣＫなどのメモリ管理アルゴリズムの適用により、キャッシュの有効利用をはかる。

具体的には、通信網管理シーケンスからのＭ－ＳＥＴないしＭ－ＧＥＴ要求に対して、要求先のＭＯに対するキャッシュ内領域へのポインタ（図７におけるcrossConnect_Cache)が存在すれば当該ＭＯのキャッシュは存在し、ポインタがＮＵＬＬならば当該ＭＯのキャッシュは存在しない。当該キャッシュが存在しない場合、マネジャ機能を実現する通信網管理システムは属性クラスＤに準じ、ＣＭＩＰコマンドを直接用いてＭＯ属性値のＧＥＴ／ＳＥＴを行う。

【００４８】一例としてＬＲＵメモリ管理が適用されている場合、キャッシュ領域が満杯になると、キャッシュ内のＭＯの属性のうちアクセスの履歴が古いものからキャッシュを消去し、対応するキャッシュ内領域へのポインタをＮＵＬＬにリセットし、領域を確保する。新しく解放された領域に、新たにアクセスされたＭＯ属性のキャッシュを確保する。

【００４９】図１２にＬＲＵを用いた場合のキャッシュ管理シーケンスを示す。図中の番号はシーケンスの実行順序に対応する。まず、「０」として、マネジャ１２がアクセスしようとするＭＯ属性がキャッシュ１５，１６中に存在するか調べ、その応答「１」がＮＯである場合、Cache 中に存在する属性データのうち、最近もっとも使われていない属性Ａ２の領域を解放する。並行してエージェント１０側へＭ－ＧＥＴを用いて当該ＭＯ属性値を取りに行き「２」、ＭＯ属性値を得、解放された領域に格納する「３」。エージェント１０側では、プロトコル（Ａ），（Ｂ）においてキャッシュ中のＭＯ属性値の更新をBroadcast にて行うため、どのマネジャがキャッシュ中にＭＯ属性を持っているか知る必要がある。このため、マネジャ１２はエージェント１０側に対して、更新されたキャッシュ状態についてエージェント１０側に通知し「４」、それに従ってエージェント１０側ではキャッシュ状態を示すテーブル（Cache Status）を更新する「５」。Cache Statusは図中に示されるように、キャッシュ可能な属性（Ａ１，Ａ２）とそれらの属性をキャッシュしているマネジャの組を表わすテーブルであり、ここでＭはマネジャのＩＤを表わす。図中に示した管理シーケンスの結果、属性Ａ１については０からＭへ変化し、属性Ａ２についてはＭから０へ変化する。

【００５０】次に、エージェントの障害復旧について、図１３を用いて説明する。同図中、マネジャ１２はエージェント１０のＭＯ４０の属性値をキャッシュ１５～１７に持っており、時刻ｔ。以前でＭＯとキャッシュとの属性値はｇで一致しているものとする。

【００５１】また、エージェント１０はＣＭＩＰを用いて通信を行っているマネジャ１２に関するアソシエーション情報を不揮発性メモリ（不揮発性記録手段）４１に記録している。このアソシエーション情報は図１４に示す如く、その時点で自エージェント１０と通信しているマネジャの名前（マネジャＩＤ）Ｍ１，Ｍ２，Ｍ３が列挙されている。

【００５２】時刻ｔ。において、エージェント１０が障害を起こし、ＭＯ４０の属性値が不定（Ｘ）になったとする。時刻ｔ，においてエージェント１０が復帰したとすると、エージェント１０はＭＯ属性値の障害前の値に復帰するために不揮発性メモリ４１を検索し、キャッシュを持っている可能性のあるマネジャＭ１，Ｍ２，Ｍ３にＣＭＩＰを用いて問い合わせ（Ｒ）を行い、各マネジャは問い合わせのあったＭＯ属性に対応するキャッシュ１５～１７を持っている場合にそのキャッシュの値（Ａ）をエージェント１０に返す。これにより、時刻ｔ，以後はエージェント１０上のＭＯ４０の属性値が障害以前の前の値（ｇ）に復帰する。以後は通常のコヒーレンシ・プロトコルに基づいて動作する。

【００５３】図１５にエージェント１０の障害復旧のフローチャートを示す。エージェントに障害が発生し、障害が回復した場合にこの処理を開始する。まず、ステップＳ９００で不揮発性メモリ４１に名前が記録されているマネジャに対してＭＯ属性値の問い合わせを行う。また、エージェント１０に不揮発性バックアップ装置が接続されていれば、これについてもＭＯ属性値を問い合わせる。次にステップＳ９１０で問い合わせを行ったマネジャ及び不揮発性バックアップ装置からの応答を受信する。ステップＳ９２０では同一のＭＯ属性についての複数の応答があった場合、そのＭＯ属性の更新時間、即ちコミット時間が最も新しいものを採用する。これは、プロトコル（Ａ）を適用する属性の場合、各マネジャのキャッシュの属性値が一時的に不一致の場合があるからである。ステップＳ９３０では採用した応答によってＭＯ４０を再現し、エージェント動作を復旧する。

【００５４】すなわち、マネジャ上のキャッシュを障害時のバックアップとして利用することにより、特別なメカニズム設けることなくネットワークの耐障害性、信頼性を高めることができる。なお、エージェント１０ではなく、マネジャが障害を起こした場合も同様であり、この場合は通常のmanager _init.cにより自動的にコヒーレンシ・プロトコルが働きキャッシュの内容が障害以前の値に復帰するので、この場合は問題でない。

【００５５】次に、本発明を階層的ネットワークに適用した場合について説明する。図１６に階層的ネットワークの構成図を示す。同図中、ネットワーク５０は複数のドメインに分割され、各ドメインがサブネットワーク５１，５２を構成している。ＮＭはネットワーク５０のマネジャ、ＳＮＭ.0，ＳＮＭ１はサブネットワーク５１，５２夫々のマネジャを表わし、Ｓ，Ｔ夫々はネットワーク終端点（ＴＴＰ）のコネクト装置であり、Ｏ，Ｐ，Ｕ，Ｖはサブネットワーク５１のドメイン境界の終端点（ＣＴＰ）のコネクト装置である。

【００５６】ここで、ネットワークの終端点Ｓ，Ｔ間のパス設定は、次の順に実行される。

（１）パス探索
（２）パス情報を表すTrail （伝送路）、ＣＴＰ、ＴＴＰ等のＭＯ生成
（３）パス導通（Activate）
（１）のパス探索ではコスト的に最適なパスを選ぶためＮＭはＳＮＭ内のパスに関するコスト情報が必要になり（Ｏ－Ｐ，Ｑ－Ｒの区間）、（２）で生成するＭＯの一部については、管理上ＮＭとＳＮＭ両方からアクセスされる。例えばＯ，Ｐ，Ｑ，Ｒなどのドメイン間境界のクロスコネクト装置のＭＯなどはその例である。この図においてＮＭ，ＳＮＭ０，ＳＮＭ１がマネジャでありＯ，Ｐ，Ｑ，Ｒがエージェントである。

【００５７】この場合、ＭＯの属性をコヒーレンシ要求に従って分割し、それぞれのクラスに異なるコヒーレンシ・プロトコルを適用する。図１７はＳＮＭによる属性更新シーケンスを表わしている。このようなシーケンスは、ＳＮＭがその管理ドメインであるサブ・ネットワークに関する管理シーケンスを実行するに当り、そのドメイン間境界にあるクロスコネクト装置のＭＯ属性値の変更を行う場合に必要となる。

【００５８】図１６の例ではＳＮＭ０がＯ，Ｐ，Ｕ，Ｖの属性値変更を行う場合、ＳＮＭ１１がＱ，Ｒ，Ｗ，Ｘの属性値変更を行う場合がこれに当る。また、図１７では属性クラスＢについてキャッシュ更新が成功する場合のみを示しているが、実際においては複数のマネジャがキャッシュを持つ場合キャッシュ更新が常に成功するとは限らない。とくに、ロック要求は複数のマネジャが同時にエージェントへ書き込みを行うことを防ぐために必要であるが、ロックが得られない場合シーケンスは失敗する。失敗の場合はシーケンスを途中で中断し、最初からやり直す。

【００５９】ここで、属性クラスＡ　クロスコネクトＭＯ属性のうち、属性クラスＡに類別したもの、パス探索におけるＳＮＭ内のパスに関するコスト情報を表わすコスト，テーブルなど。属性クラスＢ　クロスコネクトＭＯ属性のうち、属性クラスＢに類別したもの。

【００６０】図１７（Ａ），（Ｂ）の例では、ＮＭ、エージェント、ＳＮＭ間のメッセージの交換を、それぞれ属性クラスＡと属性クラスＢに分けて示している。ここでエージェントはＣＭＩＰにおけるエージェントを意味し、具体的にはＱ～Ｒ，Ｕ～Ｘなどのクロスコネクト装置を表わしている。すなわち、ＭＯ及びその属性値はそれぞれエージェント上にあり、ＮＭ及びＳＮＭはそのＭＯ属性対応のキャッシュをメモリ上に持っている場合を示している。

【００６１】シーケンス開始時（時刻 t_0 以前）のＭＯ属性値をｇとし、ＳＮＭにおいて実行中の管理シーケンスから時刻 t_0 にＭＯの属性値をｈとする書き込み（Ｖ）が発生する場合を示している。図中ＮＭ，エージェント，ＳＮＭの柱内の値（ｇ，ｈ，Ｘ）はそれぞれ各時刻におけるＮＭ上のＭＯ属性対応のキャッシュ、エージェント上ＭＯの属性値、ＳＮＭ上のＭＯ属性対応のキャッシュの値を示す。ここで、Ｘはキャッシュが無効状態にあり、読み出し不可になっていることを表わす。以下、図中シーケンスのメッセージ種別、記号、及び時刻の意味について説明する。

【００６２】ｇ，ｈ　ＭＯの属性値、あるいは対応するキャッシュの値。
Ｘ　　キャッシュ無効状態。読み出しは不可。
Ｖ　　ＳＮＭの管理シーケンスにおけるＭＯ属性の書き込み要求。
$n\tau$　　定期的にエージェント側からマネジャ側へブロードキャストが行われる時刻。

【００６３】Ｒ　　ＳＮＭからエージェントへの属性値更新要求（Update Request）。
Ｌ　　ＳＮＭからエージェントへのロック要求（Lock Request）。
Ｉ　　エージェントからＮＭへのキャッシュ無効化要求（Invalid Request）。

【００６４】Ａ　　エージェントからＳＮＭへのＬ又はＩメッセージに対応する返答（Acknow legement）。
Ｂ　　更新された新しい属性値の伝搬（ブロードキャスト）。
図１７（Ａ）において、時刻 t_1 でＳＮＭよりの更新要求エージェントに到着。時刻 t_2 で時間 τ 以下毎の定期的なエージェントからのブロードキャストによりＮＭ，ＳＮＭ内のキャッシュを更新する。時刻 t_1 ～ t_2 間ではエージェント上のＭＯと、ＮＭ，ＳＮＭ上のキャッシュとは不一致状態である。

【００６５】図１７（Ｂ）において、時刻 t_3 で、エージェントからのＩメッセージによるＮＭのキャッシュ無効化が行われる。特に、エージェントが新しい値であるｈをＢメッセージによってブロードキャストする場合、ＮＭ及びＳＮＭにおいて全てのキャッシュが無効化されていることを確認する必要がある。すなわち、Ｂメッセージの送出前にＮＭ及びＳＮＭからＡ及びＲメッセージが到着していなければならない。時刻 t_4 でＮＭ，ＳＮＭ夫々のキャッシュ更新が終了し、ＮＭ、ＳＮＭ上のキャッシュ再び有効になる。時刻 t_3 ～ t_4 間では、キャッシュ読み出し不可である。

【００６６】図１７（Ａ），（Ｂ）においては、属性クラスＡ及びＢの場合のみ示しているが、属性クラスＣ，Ｄの場合はそれぞれ自明である。すなわち、属性クラスＣの場合は定義より書き込みシーケンスは発生せず、属性クラスＤの場合は通常行われるＣＭＩＰのＭ－ＳＥＴコマンドを用いてマネジャはエージェント上ＭＯの属性値を変更する。

【００６７】このように、属性クラスＡで必要とされるキャッシュ・コヒーレンシを維持するためのメッセージ数は属性クラスＢにそれに比してはるかに少なく、また

属性クラスCの属性はコヒーレンシ維持の為のメッセージを必要としないことから、コヒーレンシ・プロトコルによるメッセージ数が１／１０程度まで激減し、管理シーケンスの実行を高速化しながら、ネットワークが輻輳状態に陥る危険を排除することが可能となる。また、以上のコヒーレンシ・プロトコルは全てＣＭＩＰを用いて実現できるので、ＣＭＩＰのプロトコル・スタックを守ったまま、すなわち現状の通信網管理システムを全く変更することなく本発明の通信網管理キャッシュ制御方式を導入することができる。

【００６８】図１８（Ａ），（Ｂ）はエージェントが時刻ｔ。に自発的に属性値を変更した場合の属性クラスＡ，Ｂ夫々のシーケンスを示している。図１８（Ａ）において、時刻ｔ。でエージェント上で属性値がｇからｈに変更。時刻ｔ，で定期的なエージェントからのブロードキャストにより、ＮＭ、ＳＮＭ内のキャッシュを更新する。時刻ｔ。～ｔ，間ではエージェント上のＭＯと、ＮＭ、ＳＮＭ上のキャッシュとは不一致状態。

【００６９】図１８（Ｂ）において、時刻ｔ，でエージェントからのＩメッセージによるＮＭ，ＳＮＭの無効化が行われる。この後、ＮＭ，ＳＮＭからエージェントへＡメッセージが到着するとエージェントはＮＭ，ＳＮＭへＢメッセージをブロードキャストする。時刻ｔ，でＮＭ，ＳＮＭ夫々のキャッシュ更新が終了し、ＮＭ，ＳＮＭ上のキャッシュが再び有効になる。

【００７０】図１９（Ａ），（Ｂ）はＮＭが時刻ｔ。に属性値を変更する場合の属性クラスＡ，Ｂ夫々のシーケンスを示している。図１９（Ａ）において、時刻ｔ，でＮＭよりの更新要求エージェントに到着。時刻ｔ，で時間τ以下毎の定期的なエージェントからのブロードキャストによりＮＭ，ＳＮＭ内のキャッシュを更新する。時刻ｔ，～ｔ，間ではエージェント上のＭＯと、ＮＭ，ＳＮＭ上のキャッシュとは不一致状態である。

【００７１】図１９（Ｂ）において、時刻ｔ，で、エージェントからのＩメッセージによるＳＮＭのキャッシュ無効化が行われる。特に、エージェントが新しい値であるｈをＢメッセージによってブロードキャストする場合、ＮＭ及びＳＮＭにおいて全てのキャッシュが無効化されていることを確認する必要がある。すなわち、Ｂメッセージの送出前にＳＮＭ及びＮＭからＡ及びＲメッセージが到着していなければならない。時刻ｔ，でＮＭ，ＳＮＭ夫々のキャッシュ更新が終了し、ＮＭ、ＳＮＭ上のキャッシュ再び有効になる。時刻ｔ，～ｔ，間では、キャッシュ読み出し不可である。

【００７２】図２０（Ａ），（Ｂ）はＳＮＭにおいて実行中の管理シーケンスにおいて読み出し要求が発生した場合の属性クラスＡ，Ｂ夫々のシーケンスを示している。時刻ｔ。において発生する読み出し要求（Ｖ）に対し、ＳＮＭ上のキャッシュの値（ｇ）が直ちに返される（Ｗ）。これは属性クラスＡ，Ｂ及びＣに対して同じであり、何となればキャッシュ・コヒーレンシ・プロトコルはエージェント上のＭＯ属性値とＮＭ，ＳＮＭ上のキャッシュの値が等しい（ｇ）ことを保証するからである。一方、属性クラスＤについては、通常行われるＣＭＩＰのＭ－ＧＥＴコマンドを用いてマネジャはエージェント上の属性値を読み出す。

【００７３】図２１（Ａ），（Ｂ）はＮＭにおいて実行中の管理シーケンスにおいて読み出し要求が発生した場合の属性クラスＡ，Ｂ夫々のシーケンスを示している。時刻ｔ。において発生する読み出し要求（Ｖ）に対し、ＮＭ上のキャッシュの値（ｇ）が直ちに返される（Ｗ）。これは属性クラスＡ，Ｂ及びＣに対して同じであり、何となればキャッシュ　コヒーレンシプロトコルはエージェント上のＭＯ属性値とＮＭ，ＳＮＭ上のキャッシュの値が等しい（ｇ）ことを保証するからである。一方、属性クラスＤについては、通常行われるＣＭＩＰのＭ－ＧＥＴコマンドを用いてマネジャはエージェント上の属性値を読み出す。

【００７４】このように、属性分割を行い、かつキャッシュを使用することにより、使用しない場合に比較して読み出し時間を大幅に短縮し、同時に読み出しに伴うマネジャ・エージェント間のＣＭＩＰのメッセージ数を大きく削減する。このように、本発明によればキャッシュを用いて管理シーケンス実行の大幅な高速化を可能にしながら、コヒーレンシ・プロトコルによるメッセージ数を激減させることが可能となり、よって通信網管理の高速化に寄与する。また、キャッシュを障害時のバックアップとすることにより容易にかつ効率良く耐障害性を向上し、通信網の高信頼化に寄与するところが大きい。

【００７５】

【発明の効果】上述の如く、請求項１に記載の発明によれば、管理主体にキャッシュを設けることにより、管理主体が実行する管理シーケンスを高速化することができ、管理情報の属性をキャッシュ・コヒーレンシ要求度に基づいてクラス分けすることによりキャッシュ管理の最適化をはかることが可能となる。

【００７６】また、請求項２に記載の発明によれば、管理情報の定義から初期化手段が自動生成され、この初期化手段によって管理主体に属性分割キャッシュが確保され、かつ、その初期化が行われ、属性分割キャッシュを使用しない装置との相互運用性を確保できる。

【００７７】また、請求項３に記載の発明によれば、管理情報の定義からインタフェース処理手段が自動生成され、このインタフェース処理手段によって属性クラス毎のコヒーレンシ・プロトコルが実行されるため、属性キャッシュ夫々に最適のコヒーレンシ・プロトコルが実行可能となる。

【００７８】また、請求項４に記載の発明によれば、属性分割キャッシュが夫々に最適のコヒーレンシ・プロトコルを用いてキャッシュ・コヒーレンシが確保された

め、ネットワークの管理トラヒックの増大を防止してネットワークの負荷の増大を抑制できる。

【００７９】また、請求項５に記載の発明によれば、属性分割キャッシュをリースト・リーセントリー・ユースト・アルゴリズムを用いて管理するため、キャッシュ領域が増大化することを避けることができ、メモリの有効利用が可能となる。また、請求項６に記載の発明によれば、不揮発性記録手段を有するため、代理主体と接続されている管理主体の情報を代理主体の障害時に確保できる。

【００８０】また、請求項７に記載の発明によれば、代理主体の障害復旧時に、これと接続されていた管理主体のキャッシュから管理情報を復旧できるため、キャッシュを代理主体の障害復旧に有効利用でき、実用上きわめて有用ある。

【図面の簡単な説明】

【図１】本発明の構成図を示す。

【図２】ＭＯ定義を示す図である。

【図３】キャッシュ生成と初期値設定のシステムフローチャートである。

【図４】ＩＤＬコンパイラの処理構成図である。

【図５】ＭＯ定義を示す図である。

【図６】ＩＤＬ定義を示す図である。

【図７】キャッシュ実現例を示す図である。

【図８】キャッシュ実現例を示す図である。

【図９】マネジャの初期化ルーチンのフローチャートである。

【図１０】エージェントの初期化ルーチンのフローチャートである。

【図１１】マネジャのキャッシュアクセスのフローチャートである。

【図１２】キャッシュ管理を説明するための図である。

【図１３】エージェントの障害復帰を説明するための図である。

【図１４】不揮発性メモリの記録形式を示す図である。

【図１５】エージェントの障害復旧のフローチャートである。

【図１６】階層的ネットワークの構成図である。

【図１７】ＳＮＭの属性値書き込みシーケンス図である。

【図１８】エージェントの属性値書き込みシーケンス図である。

【図１９】ＮＭの属性値書き込みシーケンス図である。

【図２０】ＳＮＭの属性値読み出しシーケンス図である。

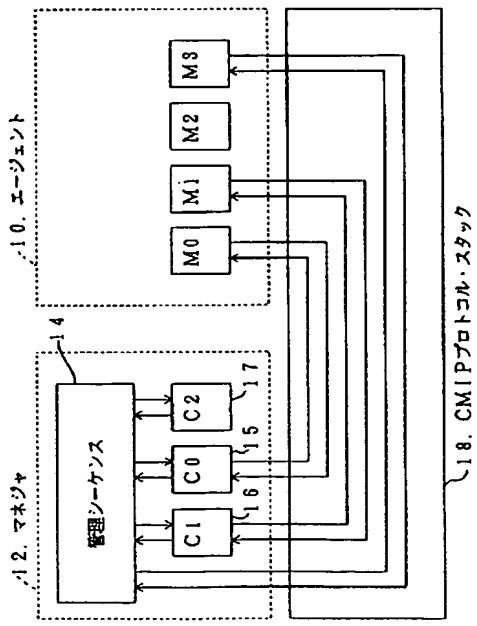
【図２１】ＮＭの属性値読み出しシーケンス図である。

【符号の説明】

１０　エージェント
１２　マネジャ
１５～１７　キャッシュ
１８　ＣＭＩＰプロトコル・スタック
２０　ＧＤＭＯ記述
４１　不揮発性メモリ

【図１】

本発明の構成図



【図６】

ＩＤＬ定義を示す図

```
S510  /*
      type''_ObjectInstance'' ''_ObjectId''
      is defined in X711 document.
      /*
S550
      typedef _ObjectInstance ObjectInstance;
      typedef _ObjectId ObjectId;

      interface crossConnection : X721_top,  MO_top  {
S600      readonly attribute ObjectId crossConnetionId;
          attribute enum administrativeState;
          readonly attribute enum operationalState;
          readonly attribute enum signalType;
          readonly attribute ObjectInstance fromTermination;
S650      readonly attribute ObjectInstance toTermination;
          readonly attribute enum directionality;
      } ;
```

【図２】

MO定義を示す図

```
S10  crossConnection MANAGED OBJECT CLASS
       DERIVED FROM "Recommendation X.721: 1992":top;
       CHARACTERIZED BY crossConnectionPackage PACKAGE
         BEHAVIOUR crossConnectionBehaviour ;
S50      ATTRIBUTES
           crossConnectionId
               GET,
           "Recommendation X.721 : 1992":administrativeState
               GET-REPLACE,
S100       "Recommendation X.721 : 1992":operationalState
               GET;
           signalType
               GET;
           fromTermination
S150           GET;
           toTermination
               GET;
           directionality
               GET;;;
S200 REGISTERED AS {m31000bjectClass 15 } ;
```

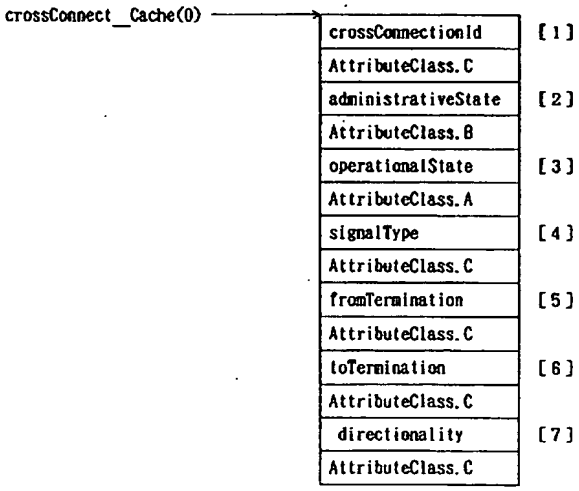
【図５】

MO定義を示す図

```
S210 crossConnection MANAGED OBJECT CLASS
       DERIVED FROM "Recommendation X.721: 1992":top;
       CHARACTERIZED BY crossConnectionPackage PACKAGE
         BEHAVIOUR crossConnectionBehaviour ;
S250     ATTRIBUTES
           crossConnectionId      -- #pragma : AttributeClass.C
               GET;
           "Recommendation X.721 : 1992":administrativeState
                                  -- #pragma : AttributeClass.B
               GET-REPLACE,
S300       "Recommendation X.721 : 1992":operationalState
                                  -- #pragma : AttributeClass.A
               GET;
           signalType             -- #pragma : AttributeClass.C
S350           GET;
           fromTermination        -- #pragma : AttributeClass.C
               GET;
           toTermination          -- #pragma : AttributeClass.C
               GET;
S400       directionality         -- #pragma : AttributeClass.C
               GET;;;
REGISTERED AS {m31000bjectClass 15 } ;
```

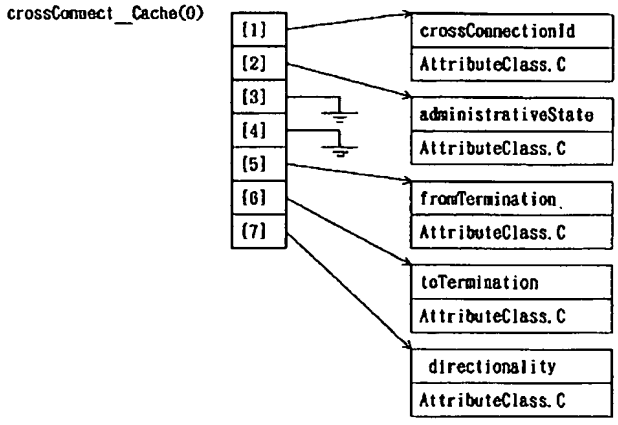
【図７】

キャッシュ実現例を示す図



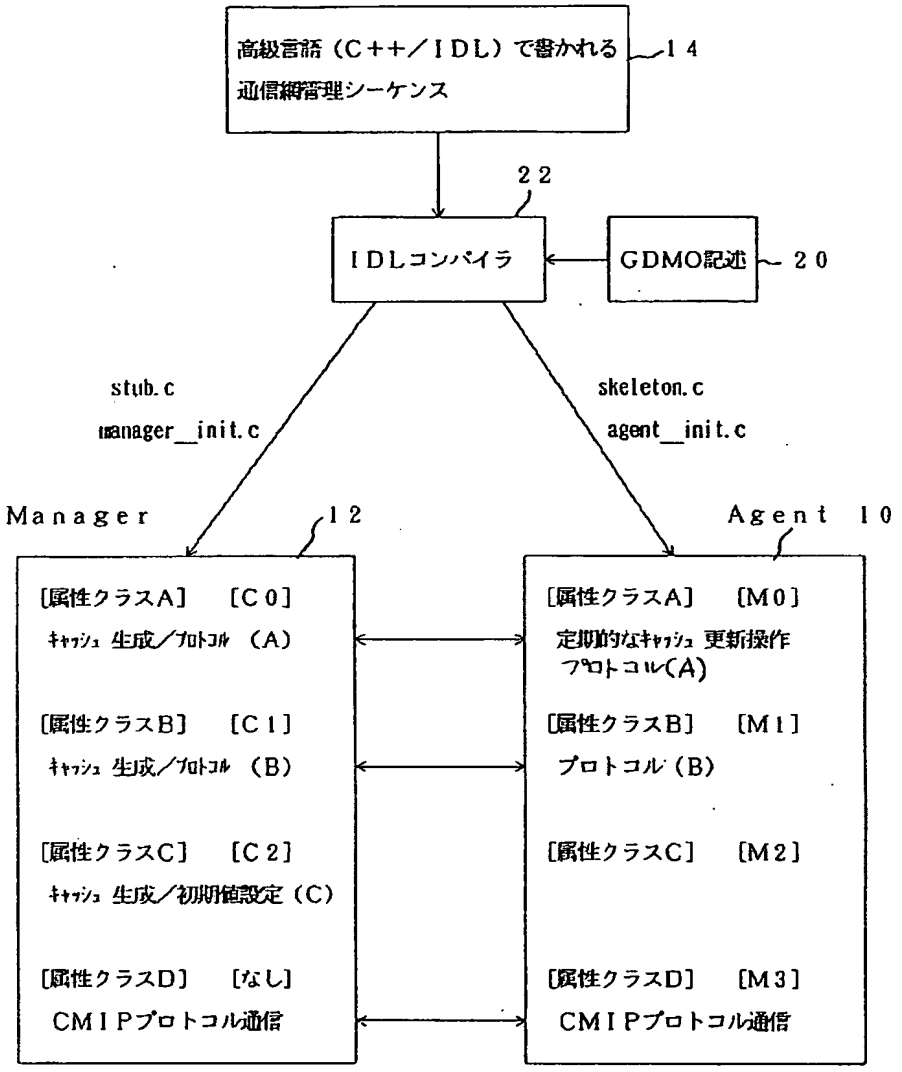
【図８】

キャッシュ実現例を示す図



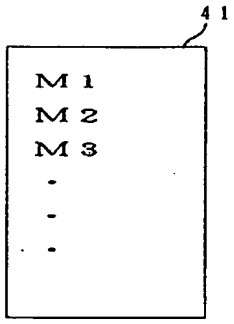
【図３】

キャッシュ生成と初期設定のシステムフローチャート



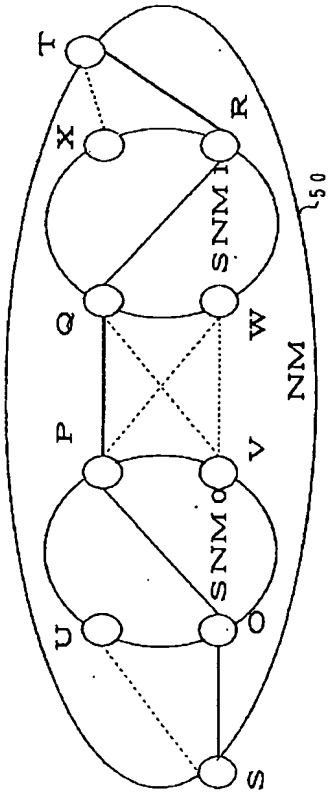
【図１４】

不揮発性メモリの記録形式を示す図



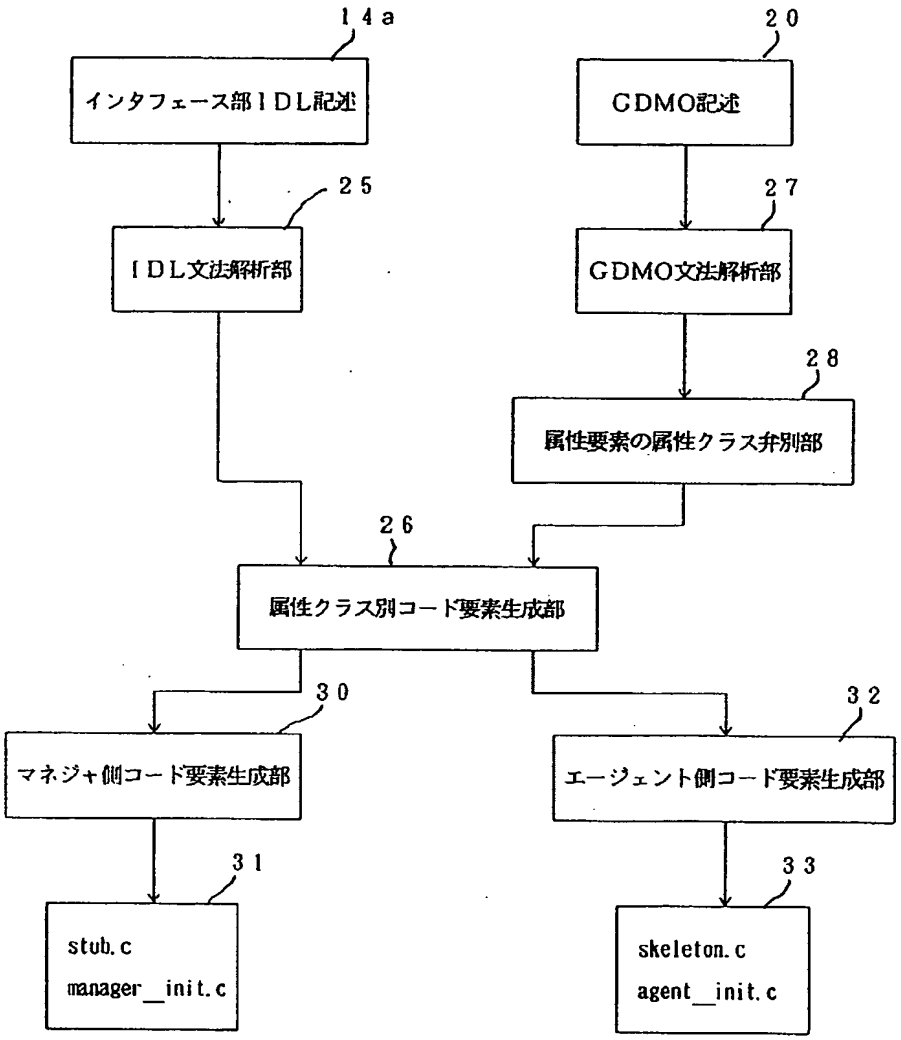
【図１６】

階層的ネットワークの構成図



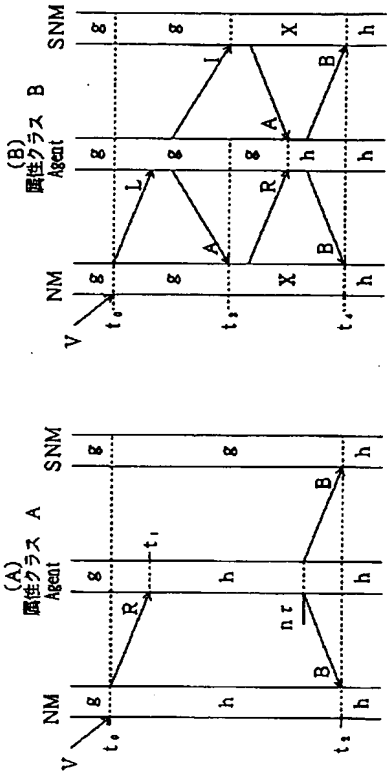
【図４】

ＩＤＬコンパイラの処理構成図



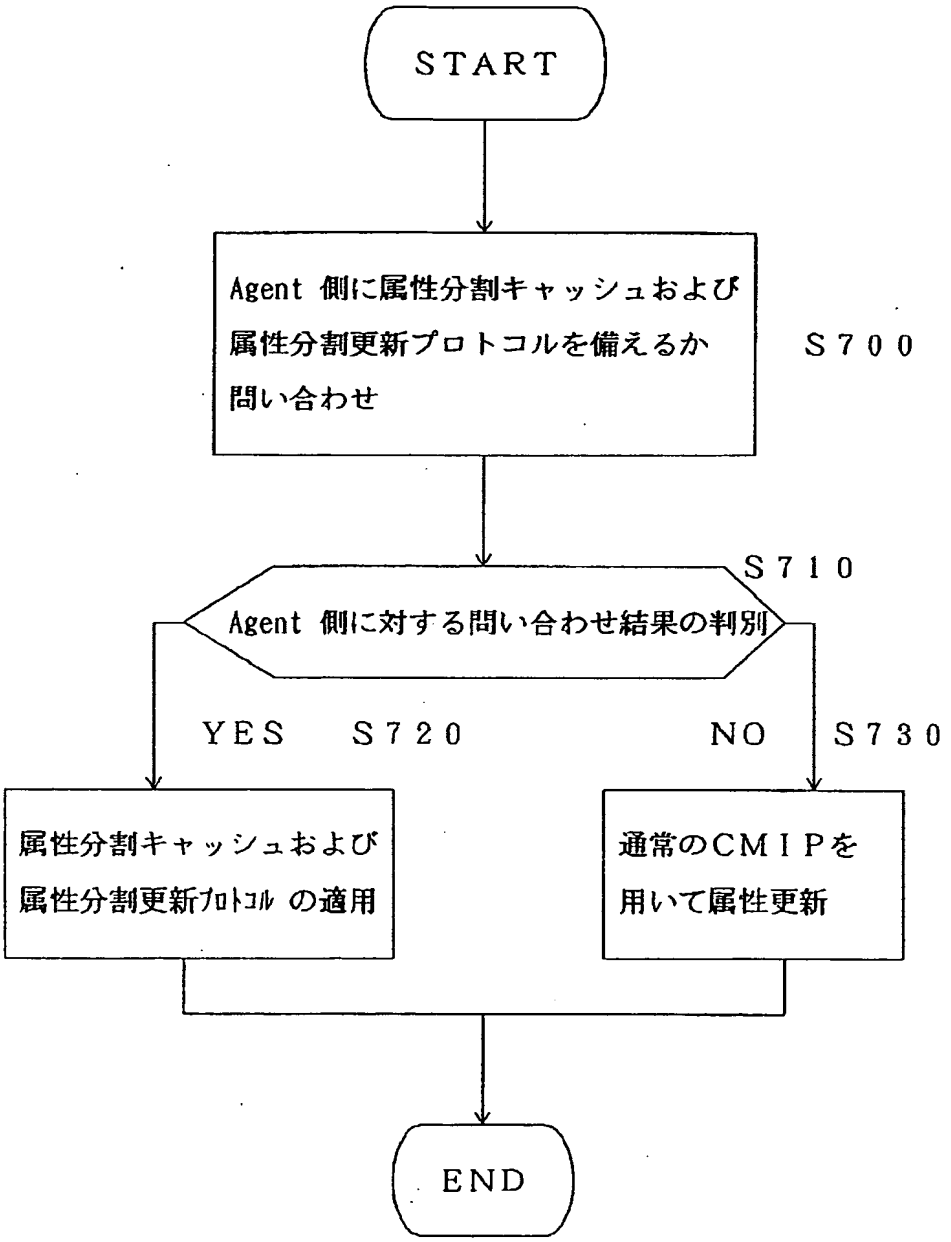
【図１９】

ＮＭの属性値書き込みシーケンス図



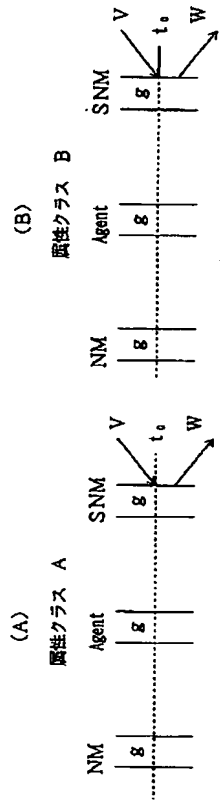
【図９】

マネジャの初期化ルーチンのフローチャート



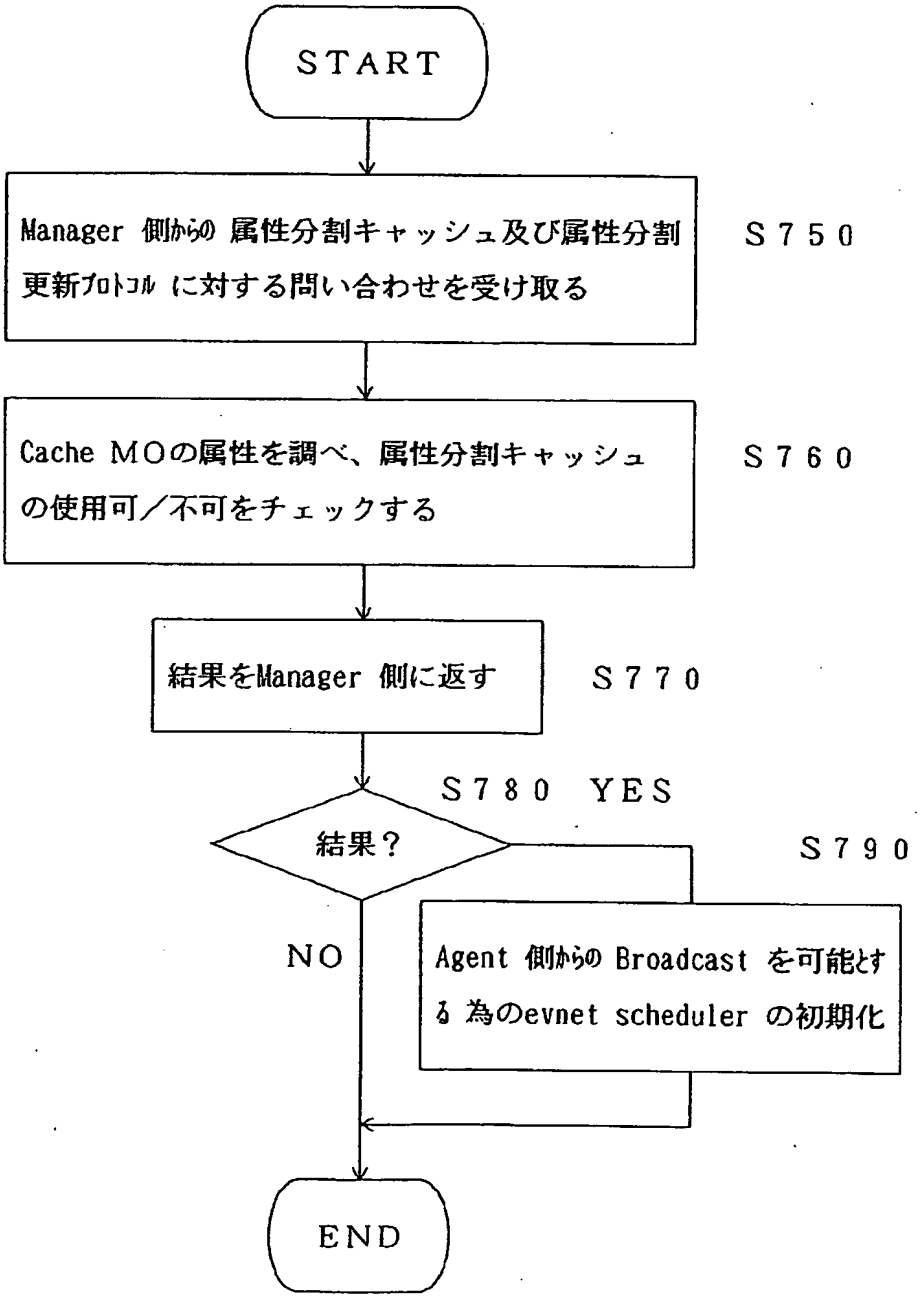
【図２０】

ＳＮＭの属性値読み出しシーケンス図



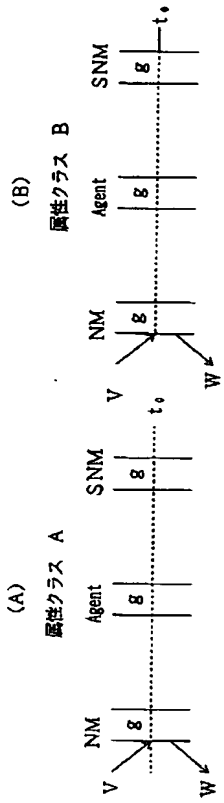
【図１０】

エージェントの初期化ルーチンのフローチャート



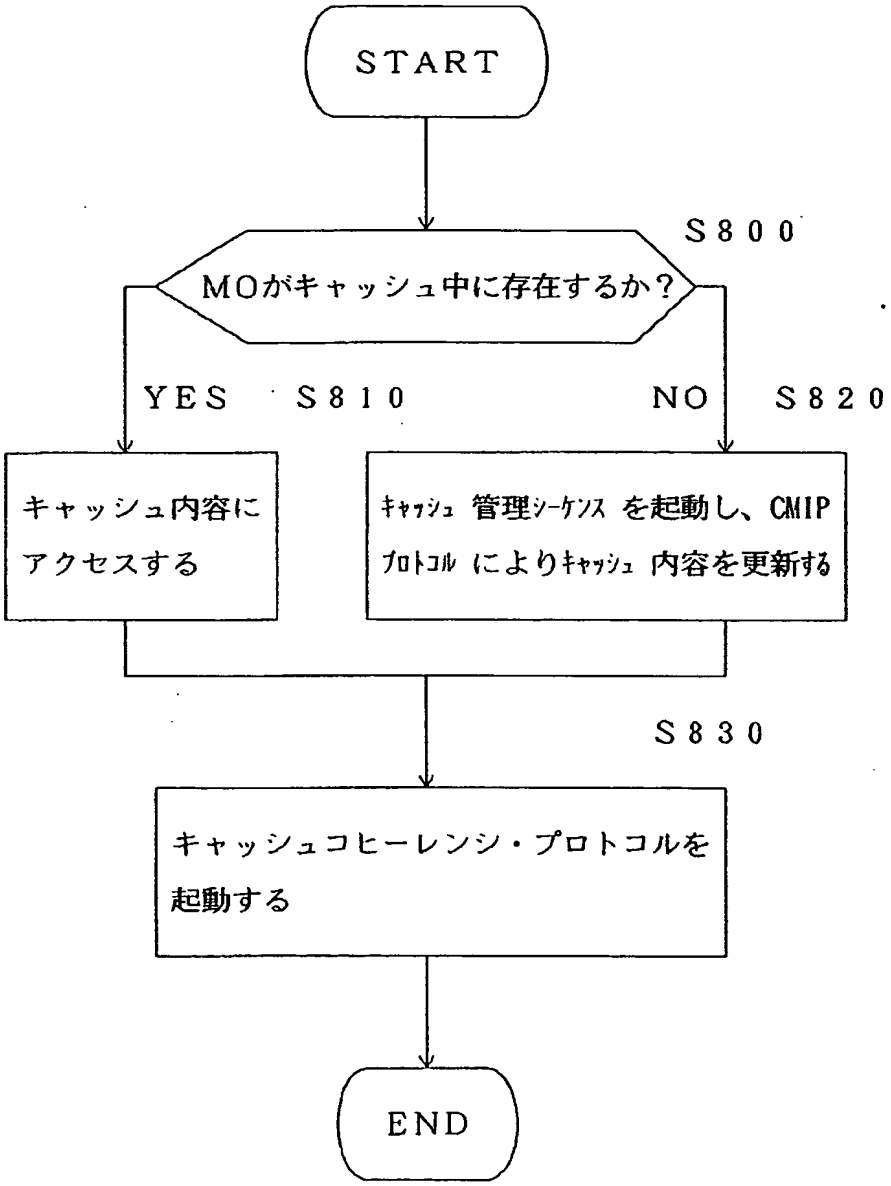
【図２１】

NMの属性値読み出しシーケンス図



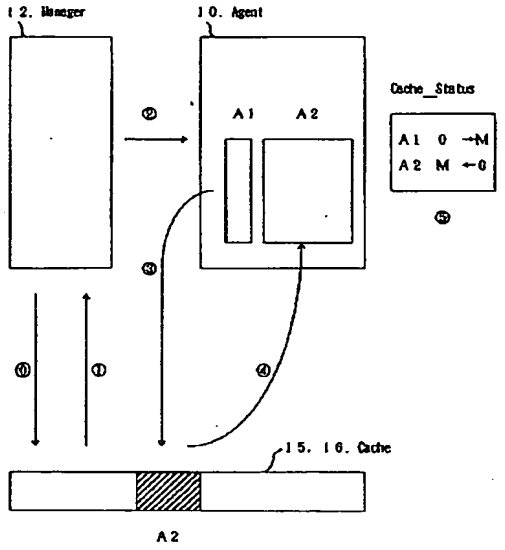
【図１１】

マネジャのキャッシュアクセスのフローチャート



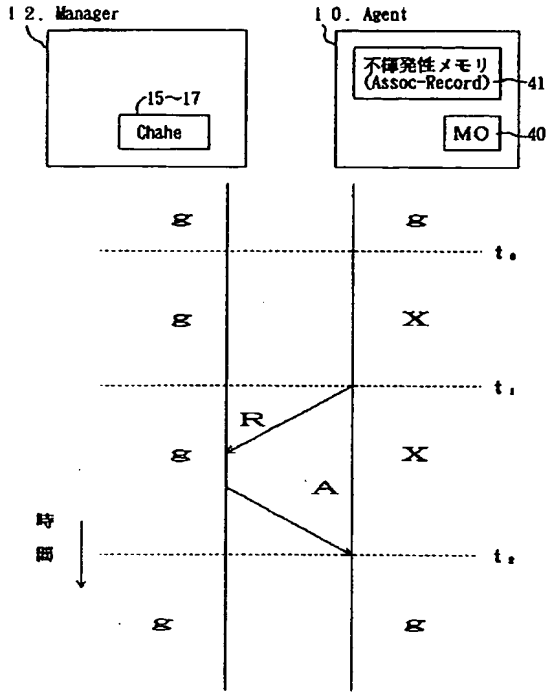
【図１２】

キャッシュ管理を説明するための図



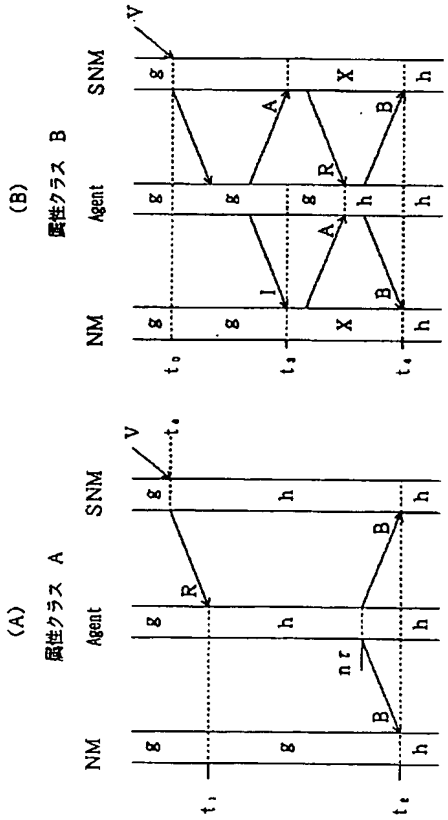
【図１３】

エージェントの障害復帰を説明するための図



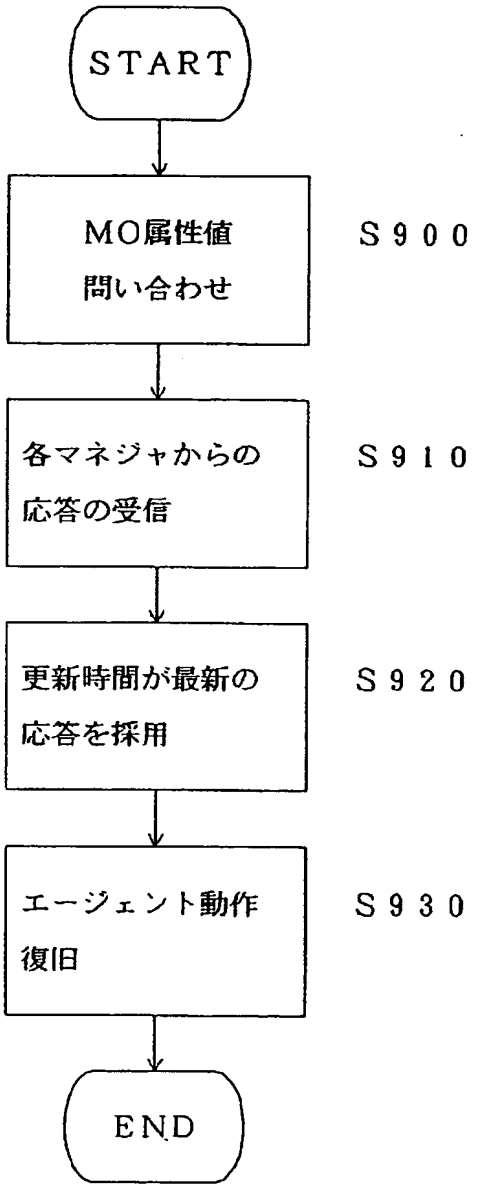
【図１７】

ＳＮＭの属性値書き込みシーケンス図



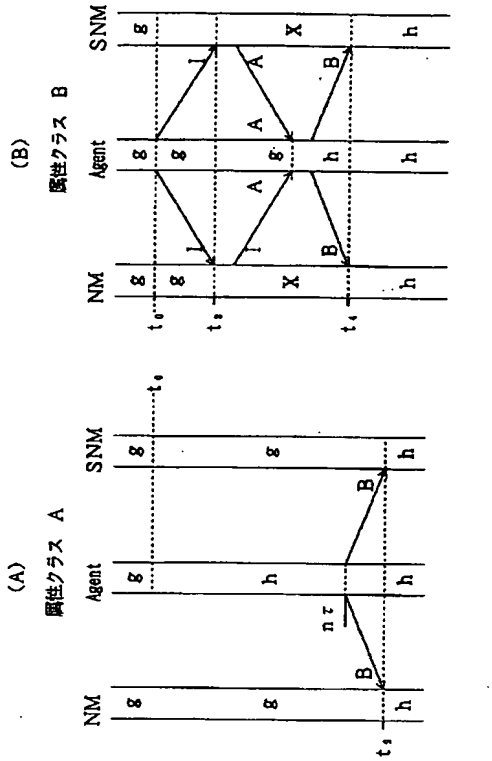
【図１５】

エージェントの障害復旧のフローチャート



【図１８】

エージェントの属性書き込みシーケンス図



フロントページの続き

(72)発明者　東　充宏
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(72)発明者　中条　孝文
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内